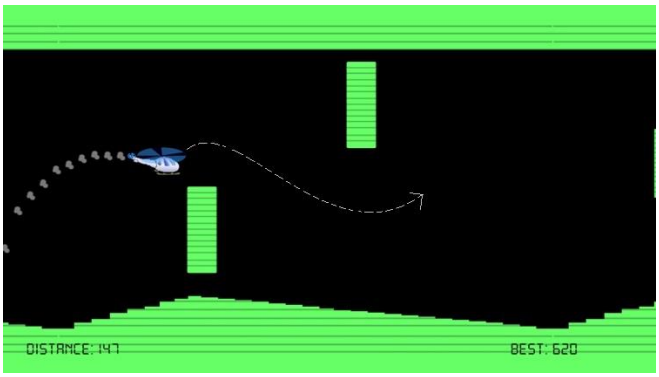
Side Scrolling UFO game using NeuroEvolution of Augmenting Topologies (NEAT) in Python

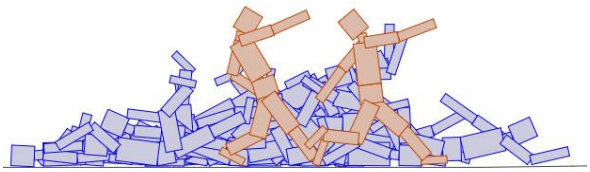
An Overview

For this project, our team looked to create and apply an artificially intelligent algorithm to play a side scrolling game. Side scrolling games have been quite popular since the 1980's. These types of games are characterized by near infinite game play where the player encounters and must avoid objects that come into the field of play. There are a couple of good examples for this type of game. One example would be the Google Chrome Dinosaur game. Another good example of this type of game would be the Helicopter game. For our project, we created a game in this style with the premise of the game being a UFO avoiding asteroids in space. To this game, we applied the NEAT (Neuro-Evolution of Augmenting Topologies) genetic algorithm in order to teach a neural network how to play this game.



NEAT is an evolutionary algorithm that creates artificial neural networks. This is an approach to solving reinforcement learning tasks as it does not require supervision. NEAT attempts to gain advantage when compared to any fixed topologies method by evolving neural network topologies along with its weights on any benchmarking reinforcement learning tasks. There is increased efficiency here due to: employing a principled method of crossover of different topologies, use of speciation to protect structure, and incremental growth from minimal structures.

When implementing this in Python, a population of individual genomes with two sets of genes is maintained to build the artificial neural network. To determine a solution to the problem, the fitness of the genome is calculated as a function of score. The greater the fitness, the greater the ability to avoid the obstacles. A generation with the candidates possessing the genetic phenotypes with the highest fitness scores, are used to create successive iteration of candidates. This incrementally improves the fitness scores of each generation and also its most successful candidates. For instance, a generation of 20 candidates that are learning to walk will have 1-2(user defined number) of candidates that will make the farthest distance. The genetic information of these candidates will then be used to create a generation of candidates that incrementally improve and walk better compared to the previous generation.

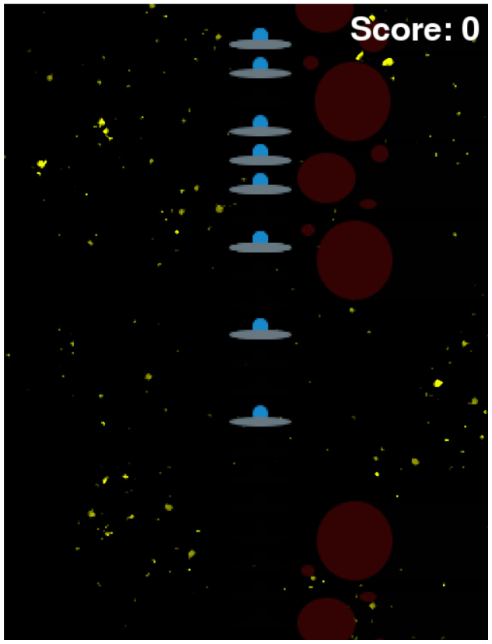


Game:

The game that was created was inspired by the Helicopter game and by Flappy Bird. In this game, there are UFOs that are trying to navigate themselves through an asteroid field. The purpose of this game is to survive and pass as many asteroid obstacles as possible. The UFO loses once it has either collided with an asteroid obstacle or if it flies off screen.

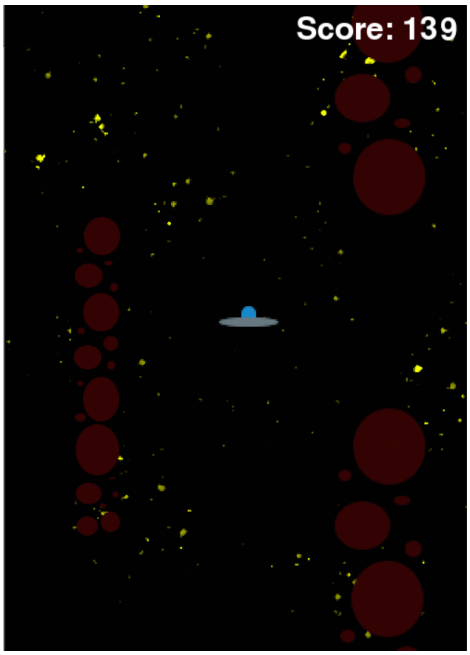
In this game, there are two types of asteroid obstacles. The first is an asteroid belt that spans the height of the screen and contains a gap that the UFO can pass through. The second is a group of

asteroids that spawns in the middle of the screen, here the UFO can either go above or below to avoid the collision. The order of these asteroid objects and their position on the screen are randomized and are different in each run of the game. This forces the player, in this case the NEAT algorithm, to decide what to do based on the changing conditions rather than simply relying on a repeated order of actions that can be taken every run.



Learnings/Trends:

When implementing the algorithm, we found that the algorithm with hyperparameter tuning, was able to play the game successfully. That being said, there were a number of trends that we discovered. One of the trends is that the UFO had a bias towards flying at the top part of the screen. As seen in the image above, the entire population group would have a bias about moving up. Most of the candidates would die and only a selected few would progress further to acquire a higher fitness score. In fact, it took a number of iterations for the algorithm to learn or to decide to go downwards. In fact, there were times where the UFO would be at the top part of the screen and the optimal move would be to move downward, however it would fly off screen which is a death scenario in the game. A higher population size per generation or higher number of generations over the neural network trained could potentially solve the problem and thus was only a matter of training candidates across generations (assuming other hyper-parameters of the NEAT configuration remained the same)that this problem would be resolved.



Conclusion:

Overall for this project, an infinite side scrolling game was created and the NEAT algorithm was applied in order to train a neural network to learn how to play the game. The game that was created was inspired by the helicopter game and was of a UFO trying to avoid asteroid objects. The NEAT algorithm learned to play the game well when given an appropriate amount of time to train. When playing the game, it was discovered that the stated mechanics of the game had an impact on the actions that the neural network occurred. It was also discovered that the neural network learned certain strategies, for example, staying at the higher end of the board. With hyperparameter tuning, the algorithm can learn how to play the game with a higher fitness level, making AI a useful tool to play these types of side scrolling games.